A Computer-Aided Design (CAD) Embroidery Font Digitizing System

Georgios Andreadis
Aristotle University of Thessaloniki/Mechanical Eng. Dept., Thessaloniki, Greece
Email: andreadi@eng.auth.gr

Efstratia Lampridou
Aristotle University of Thessaloniki/Department of Informatics., Thessaloniki, Greece
Email: lambridou@csd.auth.gr

Peter Sherar
Cranfield University/Applied Mathematics and Computing Group, Cranfield, UK
Email: psherar@cranfield.ac.uk

***Abstract*—This paper presents a Computer-Aided Design (CAD) embroidery font digitizing system that is specialized for East-Asian language character sets. The application can process any True Type font of the Windows Operating System. The system provides many operations, including editing the font's characters by setting break lines and defining the stroke curve order for each character. Following, Bézier curves are used in order to describe the shape of the character which the user can further edit. The final outcome of the software is an accurate contour of the selected characters that can be exported in various formats in order to be fed to an Embroidery Designing System. The area of study of the thesis lies in the Computer-Aided Design field and specifically in the Computer-Aided Design for Embroidery.**

***Index Terms*—cad, design, embroidery font**

I. Introduction

Embroidery can be defined as "the art or handicraft of decorating fabric or other materials with needle and thread or yarn". The word 'embroidery' is a Middle English word derived from the old French 'broder' meaning edge or border. The art of embroidery goes back into history and its origins can be traced down to the Iron Age. Samples of embroidery are found in Ancient Egypt, Northern Europe, Persia, India and China.

A. Machine Embroidery

Although traditional hand embroidery exists for thousands of years, 'machine embroidery' is only aged at about 200 years. Josue Heilmann created the first hand-embroidery machine in 1828. This machine was able to utilize up to 4 hand-embroiderers and signalled the start of the revolution in embroidery, with many other machines soon to follow.

Today, embroidery machines can be single-head or multi-head, which can fit up to 56 heads. Each head can fit one thread colour, which means that the number of heads determine the amount of colours they can be included into the embroidery without interrupting the machine function. They also have multiple needles and the speed of the machine can be up to 1000 stitches per minute. The machines are accompanied with many extra features. Some of the most common features include LCD touch screen, user interface, network capability and USB connection, and internal memory that can store stitches and locations. Along with the embroidery machines, design software specifically to serve the embroidery requirements was developed. This break-through boosted the embroidery production and a new revolution began in the embroidery industry.

B. Computer-Aided Design

Computer-Aided design –CAD– can be defined as the use of computer technology for the design of objects, real or virtual to achieve precise drawing. The resulting drawing contains symbolic information which can be the materials, the processes, the dimensions, and the tolerances, according to application-specific conventions. Computer-Aided design may be used to design curves and figures in two-dimensional space –2D; or curves, surfaces, and solids as three-dimensional objects –3D.

CAD is an important industrial tool used extensively in various applications. Some of the applications are automotive, shipbuilding, and aerospace industries, industrial and architectural design, prosthetics, and many more. CAD is also used widely to develop computer animation for special effects in movies, advertising and technical manuals.

Computer-Aided is also utilized by the embroidery industry so as to improve the embroidery designs.

1) Computer-aided embroidery

The embroidery design software that has been developed fulfils the needs of the home, commercial and the industrial embroidery.

Manuscript received October 10, 2013; revised February 10, 2014.

doi: 10.12720/joace.3.1.15-22

Nowadays, there exist various commercial and free software products to choose from, that enable the designing and editing of embroidery patterns and images. The purpose of the design embroidery software is to translate the drawing made in the computer into stitches executed by the embroidery machine.

The software products usually have advanced user interfaces with sophisticated functions. The user can create a drawing, set up the number of stitches, define the type of the stitch (satin, run fill stitches, etc), select colours and edit the design. More complicated functions include the import of images to the software and their digitalization offering all the main attributes of the image processing.

Furthermore, some of the embroidery design software solutions offer lettering functions. These functions aim to convert automatically a *true type font* into stitches. In this way, a font character can be embroidered.

In Fig. 1, the Latin character 'a' is displayed in its final form after its processing through the CAD embroidery software EOS [1].

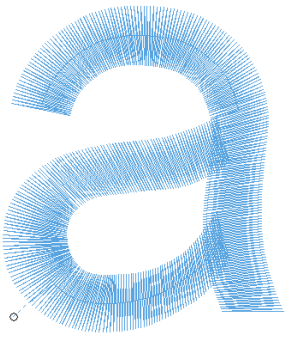

Figure 1. Embroidery design

The final embroidery result can be saved into formats that are then capable of being imported into an embroidery machine.

2) File formats

Embroidery file formats are various. Two main categories can be distinguished; the source formats and the machine formats.

Source formats files are files supported by the design software whereas machine formats files are oriented to a particular brand of embroidery machine. These files contain primarily stitch data as well as machine functions. The main drawback of the machine files is that they cannot be easily scaled or edited.

Each different embroidery machine manufacturer has developed its own embroidery machine formats specifically designed to service its brand or a specific machine. However, although machine formats where initially designed to service only their own machine brand, some formats have dominated and became so prevalent that gradually they are becoming regarded as standard even by companies in competition.

II. Describing Shapes

Shape description is a field with significant research activity. The purpose is to achieve an approximation of a given shape using segments of defined curves. Most of the algorithms developed use the Bézier curve format to perform the shape description.

A. Research Work

The research in this field is conveyed on the identification of the outline of a given shape and its description with a certain model of curves –eg. Bézier curves.

Cinque *et al* [2] in their work create a model of the object with approximate segments of the shape by using the Bézier cubic curves which interpolate most accurately the endpoints of the outline of the shape. Sohel *et al* [3] developed a shape discriptor using a Bézier Curves algorithm. The algorithm divides the shape into segments and identifies the control points.

In this project shape description is needed for the representation of the font characters. Each character is described by a set of Bézier Curves as it will be explained thoroughly below. There exists research activity on the shape description of character letters in particular.

Plass and Stone [4] developed a method for fitting shapes defined by a discrete set of data points with parametric piecewise cubic polynomial curves. Their work focuses on letter-form shapes and converts the bitmap representation of the character into curve outlines.

Sarfraz and Khan [5] developed an algorithm for capturing automatically non-Latin characters such as the Arabic ones. They also use parametric cubic Bézier curves for this task. The character is divided into a number of segments and the control points along with other significant points are detected.

Furthermore, Itoh and Ohno [6] track the outline of fonts from scanned characters' images. The algorithm extracts the contour points from the character and divides the points into a number of segments at the corner points. Following it fits a piecewise cubic Bézier curve to each segment.

Finally, Yang *et al* [7] used Bézier curves to describe the outline of Chinese letters. Firstly, the contour segment is extracted and points with high curvature are identified as corner points, and afterwards the contour segment is described by a straight line or by a Bézier curve.

In the following section the fundamental theory of curves is explained focusing on the Bézier curves as these are used by the tool developed in order to describe the outline of the font characters.

B. Curves

Graphic systems and Computer-Aided Design software use certain basic primitives in order to represent the lines. Therefore, complex shapes can arise by using these primitives. However, as the shapes become more complex, more detailed data is required to represent them accurately.

A case of such a shape is a curve. A curve is a kind of shape that can be difficult to describe. A curve can be viewed as the union of short segments of straight lines. The representation of a curve can be explicit, implicit or parametric.

An explicit curve is one defined by an equation. In general, an explicit curve in 2 dimensions is defined by

an equation of the form $y = f(x)$. The order of the polynomial $f(x)$ defines the order of the curve; therefore there are quadratic, cubic, quartic curves, and so on.

Implicit curves are those described by an implicit function, such as in 2D, $f(x,y)=0$.

Concerning parametric representation of curves, two main categories can be distinguished; approximation curves and interpolating curves. Approximating curves are those passing 'close' to the points that were used to define it whereas interpolating curves are those curves passing through the points used to define it.

Another specific kind of curve that is used extensively in computer-aided design and in computer graphics is the splines. Splines are curves with certain shape defining attributes and are especially useful as they can be utilised for modelling arbitrary functions.

Well-known and widely used splines in the field of computer-aided design are the Bézier curves, B-splines – which are the generalization of Bézier curves, and non-uniform rational B-splines (NURBS).

C. Bézier Curves

Bézier curves are used extensively in computer graphics and are constructed as a sequence of Bézier segments. [8]

A single segment cubic Bézier curve consists of 4 points, 2 end control points and 2 middle control points as depicted in Fig. 2. The end points are p_0 and p_3 while the middle control points are the p_1 and p_2. As it can be seen, the line connecting the end control point with the corresponding middle control point is tangent to the curve. The control points affect the shape of the curve. Should a control point be moved, the shape of the Bézier curve will change accordingly.

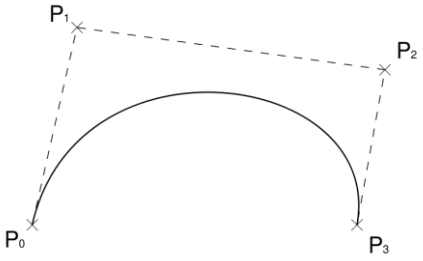

Figure 2. Bézier curve

Generally, given a set of n+1 control points P_0, P_1,….., P_n, the resulting Bézier curve of degree n is given by the (1),

$$C(t) = \sum_{i=0}^{n} P_i B_{i,n}(t) \qquad (1)$$

where $t \in [0,1]$ and $B_{i,n}$ is a Bernstein polynomial. Bernstein polynomials are defined by the formula (2),

$$B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i} \qquad (2)$$

The degree of the curve is equal to the number of control points minus 1 (n). The degree of the Bernstein polynomials is also n. The Bernstein polynomials of degree n form a *basis* for the power polynomials of degree n.

All basis functions are non-negative. Moreover, the sum of the basis functions is 1, and since they are nonnegative, it can be seen that the value of any basis function is in the range of 0 and 1.

In Fig. 3 the plot of the Bernstein polynomials of degree 9 is presented.

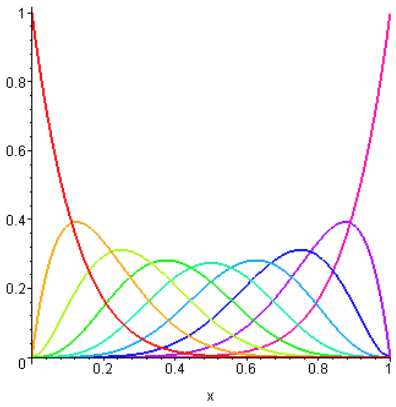

Figure 3. Bernstein Polynomials

a) Properties

The Bézier Curves have some significant properties that differentiate them from the other types of curves. A property of the Bézier curve –that has been already mentioned, is that the curve always passes through the first and last control points. However, Bézier curves do not interpolate the interior points but only approximate them. So unlike other types of curve, the interpolation is made for only some control points.

The variation diminishing property of the Bézier curves is that no straight line intersects a Bézier curve more times than it intersects its control polygon. A control polygon is the join of the line segments that pass through the control points of the curve. The importance of this property lies to the fact that the complexity of the curve (by means of turning and twisting) is not greater than the complexity of the control polygon. Therefore, the control polygon twists and turns more frequently than the Bézier curve does.

The Bézier curves also satisfy *the convex hull* property. This property declares that every point of the curve is inside the convex hull that corresponds to the curve, meaning that the curve lies completely in the convex hull. The property ensures that the curve will never pass outside of the convex hull formed by the control vertices and so it lends a measure of predictability to the curve. This property is important because it secures that any Bézier curve will be in a well-defined and computable region.

An undesirable property of Bézier curves is their *numerical instability* for large numbers of control points. This can be avoided by smoothly joining together low-order Bézier curves, as it will be explained below. Furthermore, the fact that by moving a single control point the shape of the curve is affected in whole is another negative aspect.

b) Piecewise bézier curves

More than one Bézier segment can be joined together forming a larger connected curve; also known as *piecewise Bézier curve*. Therefore, in order to create a larger curve, instead of increasing the degree of the curve –which will make the curve much more complex to calculate, cubic Bézier curves can be joined together. As already explained, a cubic Bézier curve has 4 points. The one end point of a curve can be connected with an end point of another curve forming a joint. This point, which is shared by the two curves, is called knot. The result is a curve, which if divided into pieces, cubic curves arise. Moreover, each control point only affects a specific part of the curve and not the whole curve.

Hence, rather than use a high degree curve to interpolate a large number of points, it is more common to break the curve up into several simple curves. For instance, a large complex curve could be broken into cubic curves, resulting in a piecewise cubic curve.

For the entire curve to be smooth and continuous, it is necessary to maintain C^1 parametric continuity across segments. Therefore the position and tangents should match at the end points. However, it is advised to maintain the C^2 continuity too. The way that continuity is determined is explained in the following section.

D. Continuity

Two types of *continuity* can be distinguished when two curves are joined together; *geometric* continuity and *parametric* continuity. The continuity of a curve at a knot describes how the curves meet at this point and determines the *smoothness* of the curve.

A curve or surface can be described as having G^n continuity, with *n* being the increasing measure of smoothness. Considering the segments at either side of a point on a curve the following options emerge:

- G^0: The curves touch at the join point.
- G^1: The curves share a common tangent direction at the join point. The first derivatives at the common point are proportional at this point.
- G^2: The curves share a common centre of curvature at the join point. Hence, the first and second derivatives are proportional at the point.

Parametric continuity is applied to parametric curves. It describes the smoothness of the parameter's value with distance along the curve. The types of continuity are explained below:

- C^0: The curves are joined.
- C^1: The first derivatives at the join point are equal.
- C^2: The first and second derivatives at the join point are equal.
- C^n: The first n derivatives at the join point are equal.

As can be seen, geometric continuity requires the geometry to be continuous, whereas the parametric continuity requires the parameterization to be continuous too. Having parametric continuity of a given order implies geometric continuity of the same order, yet the reverse is not valid.

Below is explained the nature of the joint –continuity– that can arise from the connection of two curves.

Fig. 4 shows the case where no continuity exists between the curves. The two curves don't share any common point. This is declared as *no continuity*.

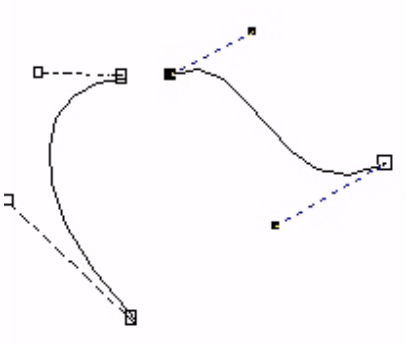

Figure 4. No continuity

In Fig. 5, the two curve segments share a single common point. In this case, the curves are connected, but with an obvious break in direction. The derivatives (for right and left segment) are different both in direction and size in this point. This is an example of *continuity zero*, C^0.

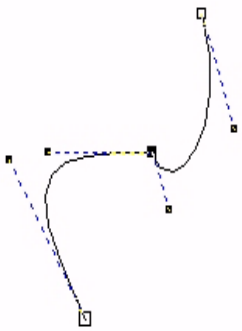

Figure 5. Continuity C^0

In Fig. 6 are presented two curve segments that have a point in common and the joint is smooth. The derivatives in this point have the same direction. The first derivatives are equal. This is defined as tangential continuity or *continuity one* – C^1.

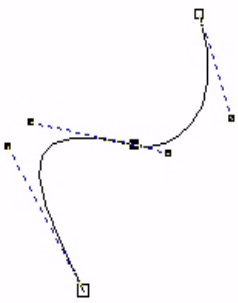

Figure 6. Continuity C^1

Finally in Fig. 7, the two curve segments have a common point, and the joint is smooth. The derivatives in the common point have both the same direction and size. In this point the curves have identical curvature. This is declared as curvature continuity or *continuity two* –C^2. Here, the second derivatives are equal.

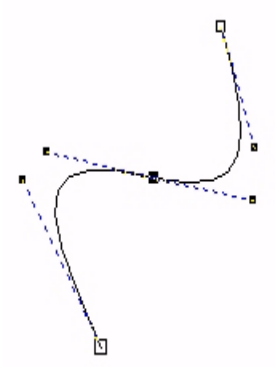

Figure 7. Continuity C^2

In the above example it is obvious that the corresponding geometric continuity exists as well.

It should also be noted that as tangential continuity implies positional continuity, whereas curvature continuity implies both tangential and positional continuity as well. Moreover, the order of a curve determines the maximum continuity possible. Thus, it might be needed a higher order of curve if more continuity is necessary. The maximum continuity that can be achieved for a given curve is *order – 2*. For instance, for the cubic curves, the maximum continuity is C^2.

E. The Branches

Most of the software solutions use Bézier Curves in order to depict the resulting areas for embroidery.

A customised model that was used in the design software EOS is the combination of two piecewise Bézier curves. From now on, this will be referred to as *branch*. A branch is defined as two piecewise cubic Bézier curves connected together. Both Bézier curves have the same number of control and end points. Each end point of the Bézier curve is connected with the corresponding end point of the second Bézier curve with a straight line named the *reference line*. Hence, each branch defines an area which will be embroidered on the same way. Moreover, the reference lines are used as a delimiter for the path that the stitches will be created on. In particular, the stitches will go perpendicular through the reference lines. An example of a branch is pictured in Fig. 8.

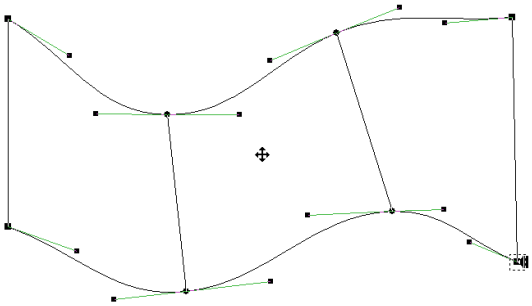

Figure 8. A branch

The continuity on the curves of the branch can differ. The different types of continuity are distinguished by the nature of the point that the two curves have in common. In particular a knot can be a corner or a non-corner. Defining a point as a corner, it means that the joint doesn't have to be smooth. In this case the resulting continuity is G^0. Otherwise, a non-corner knot creates a smoother nature of join. The tangents of the control points have the same direction but not the same size. This results in continuity G^1. A corner knot is displayed with a square, whereas a non-corner knot is displayed as a circle.

Furthermore, the calculation of the knot can be automatic or non-automatic. An automatic calculation assumes that the calculation was made by internal rules whereas the non-automatic type arises by the movement of a control point by the user.

The filled circle, as depicted in Fig. 9, is used to declare that the point that joins the two curves is not a corner and is created automatically.

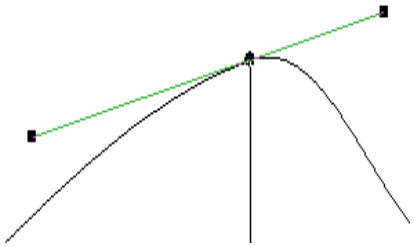

Figure 9. Non-corner, automatic knot

In Fig. 10 is shown a blank circle which defines a non-corner knot. A modification to the control points was made resulting in a non-automatic point.

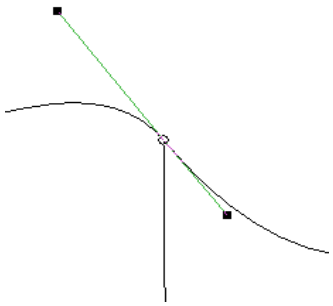

Figure 10. Non-corner, non-automatic knot

Fig. 11 shows the corner point. It is obvious that there is a break in the direction. The geometric continuity is G^0 as the derivatives are different in size and in direction. The direction of the derivatives is in the direction of the adjacent user points.

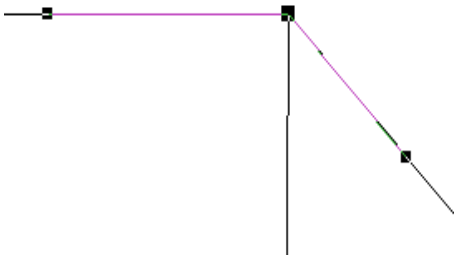

Figure 11. Corner, automatic knot

Fig. 12 shows the type of join pictured with a black square. This also declares a corner point but this one isn't created automatically. The movement of the control points is more flexible and the direction of the tangents isn't pointing to the adjacent knots.

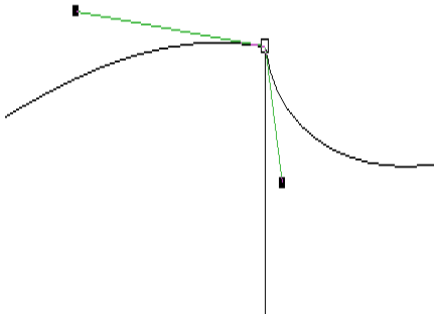

Figure 12. Corner, non-automatic knot

III. Proposed Approach

The proposed approach includes the digitization of True Type Fonts of the Windows Operating System. The outcome produced by the system is the accurate contour of a character set. The contour is described with Bezier Curves.

The reason for selecting the Bezier Curves to perform the shape description is because they have desired properties and thus have been used extensively for CAD purposes and have dominated in the font outline representation area using either quatradic Bezier Curves or cubic Bezier Curves. One advantage these curves offer is that they can be easily scalable and also they can represent the outline of the characted with accuracy.

There are three basic operations provided by the system, the setting of the break lines, the definition of the stroke order and the editing of the resulting outline. The desired result is an accurate contour of the shape of the character that will be described with the Bezier Curves. For this reason, the character should be divided into separate regions so as to decrease its complexity. Following, in each region is assigned a flow arrow declaring the embroidery direction. Moreover, each region is described with two pairwise cubic Bezier Curves that an index number that declares its ordering.

A. Break Lines

The break lines are straight lines which are used for the division of the character into separate regions. The system enables the construction of break lines into the character's bitmap. The resulting divided areas are desired to be easily defined shapes. Therefore, a complex shape can be represented as a union of simple shapes. For the East-Asian language sets this can be turn out to be essentially useful as the complex characters can be divided into segments that their contour can be easily defined with Bezier Curves.

B. Stroke Order Lines

The stroke order lines are actually arrows that can be set to each divided region. The Stroke Order Lines should be as many as the distinct areas resulting by the requested divisions. The stroke order lines define the direction that the embroidery machine will follow when embroidering the particular region. In East-Asian language characters, this is highly important information and it should comply with the writing rules of the language in order for the character to be embroidered correctly and be unambiguous.

C. Outlines

The outlines are the pairwise Bezier Curves that describe the contour of each divided region. They are calculated by the system based on the shape of the character, taking into account the break lines that were set to it. The Bezier Curves that represent the contour of the character can be further adjusted so as to best fit the character. This can be done by moving the control points or the knots and by inserting and deleting knots.

One more important parameter on the editing of the characters is the ordering. In each devided region is assigned an index number. The reording operation enables the change of the sequence of the regions. The final ordering of the letter is highly important when the character will be given to the embroidery machine. As the thread is continuous (non-stop) it is crucial that the sequence can be supported by the embroidery machine. Finally, the ordering can have a visual impact on the resulting embroidery as incorrect order may alter the desired outcome.

IV. Case Study

A case study is presented demonstrating an example of the designing system. The case study was applied to a Japanese character with ASCII code 38550. The character is shown in Fig. 13.

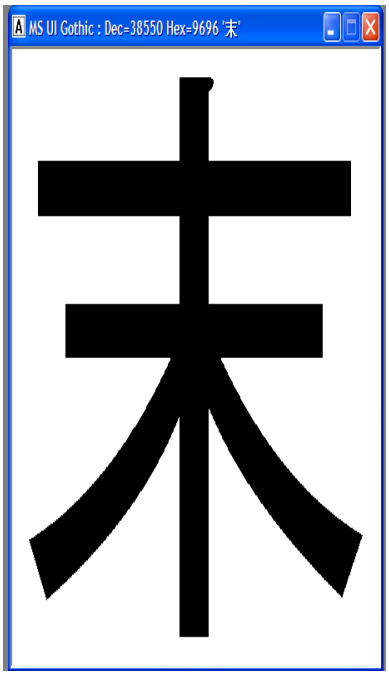

Figure 13. Japanese character

Firstly the break lines are defined on the character in order to divide the character to separate areas that split

the character into distinct regions. Each region has the same embroidery properties.

Following, the stroke order lines are set to the character. One line per region is set. The stroke order lines refer to the direction of the embroidery depending on the writing rules that apply to the language.

In Fig. 14 the result of the two first steps is depicted. The break lines are shown in blue color, whereas the stroke order lines are presented with yellow. In red is pictured the active object –which in this case is a stroke order line. The active object is the one that is selected by the user, and therefore it can be edited.

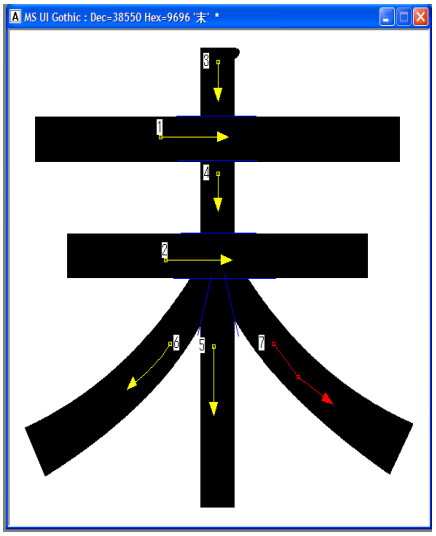

Figure 14. The break lines and the stroke order lines

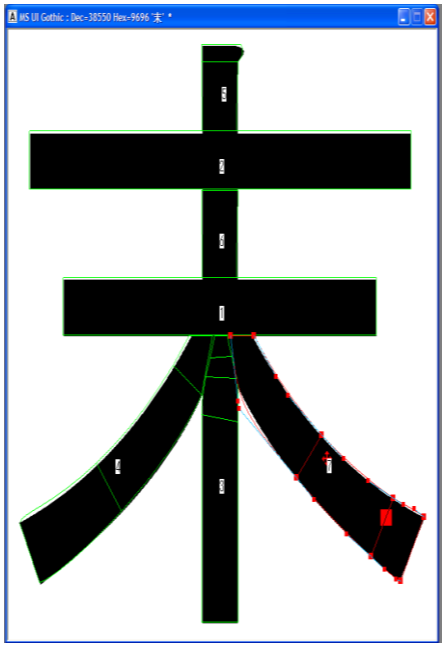

Figure 15. The outlines

The final step is the calculation of the Bezier Curves that correspond to each region. The user can further edit the Bezier Curves in order to best fit the contour of the character. In Fig. 15, the outline of the characters described with Bezier Curves is depicted with green color. Again the active object is pictured with red color.

V. Conclusions and Future Work

In this paper is presented a Web-based design Framework for Shapes Outline which is used for the digitization of true type font characters. This application is especially useful for East-Asian language characters as the way those characters are embroidered is crucial in order for them to be legible.

The system enables the process of any True Type Font available on the Windows Operating System and following the editing of the characters so that the resulting outline best fits the contour of the character in hand. The application is easy to use by users with the requirement of knowing the writing rules of the characters that are processed and the fundamental rules of the embroidery procedure. The system has the desired functionality and the user interface is friendly and direct.

Furthermore, complex character sets such as those of the East-Asian Languages can be embroidered with improved quality.

In the future the application could be further extended in several aspects. Firstly, the incorporation of additional information regarding the embroidery design is an important extension that would make the application a stand-alone one, as the resulting outcome would be directly imported into the embroidery machine with no further processing. Such information is for instance the type, the position and the number of stitches. The result would be a complete font system providing all the required functionality for the embroidery process of a font.

A further extension, which can be significantly useful, is the deployment of a Web Service that would communicate with the application in a client-based architecture that could provide remote access to it through the World Wide Web (e.g. SOA).

References

[1] EOS. [Online]. Available: http://www.eos3.com

[2] L. Cinque, S. Levialdi, and A. Malizia, "Shape description using cubic polynomial bézier curves," *Pattern Recognition Letters*, vol. 19, pp. 821-828, 1998.

[3] P. Michael and S. Maureen, "Curve-fitting with piecewise parametric cubics," *Computer Graphics*, vol.17, pp.229-239, 1983.

[4] S. F. Ahmed, G. C Karmakar, and L. S Dooley, "An improved shape descriptor using bézier curves," *Lecture Notes in Computer Science*, vol. 3776, pp. 401-406, 2005.

[5] M. Sarfraz and M. A. Khan, "Automatic outline capture of arabic fonts," *Information Sciences*, vol. 40, pp. 269-281, 2002.

[6] I. Koichi and O. Yoshio, "A curve fitting algorithm for character fonts," *Electronic Publishing*, vol. 6, pp.195-205, 1993.

[7] H. M. Yang, J. J. Lu, and H. J. Lee, "A bézier curve-based approach to shape description for chinese calligraphy characters," in *Proc. Sixth International Conference on Document Analysis and Recognition*, 2001, pp. 276-280.

[8] B. Pierre, *Numerical Control: Mathematics and Applications*, John Wiley & Sons Ltd, 1972.

Georgios Andreadis was born in Serres, Greece, 5 November 1959. He received his PhD in 1997 from the Mechanical Eng. Department of Aristotle University of Thessaloniki, Greece. His major field of study is CAD/CAM.

He is an Assistant Professor in the Mechanical Eng. Department of Aristotle University of Thessaloniki, Greece. He is author of a number of research papers dealing with Web Based CAD/CAM. Current and previous research interests are: CAPP, Cloud Manufacturing.

Professor Andreadis is a member of the Technical Chamber of Greece.

Efstratia Lampridou graduated from the Department of Informatics of Aristotle University of Thessaloniki, Greece. In 2010 she received her MSc in Informational Systems at Informatics Department of Aristotle University of Thessaloniki, Greece.

Peter Sherar holds degrees in Mathematics from the Universities of London, Warwick and Cranfield. His principle research interest is computational geometry and topology with applications to curve and surface modelling and geometric and solid modelling.
He joined Cranfield as a research officer in 1987 and has worked with a number of companies on the implementation of computer algorithms for geometric design and data translation. He is author of a number of research papers and books dealing with the mathematical and computational aspects of product modeling.